/\* Arduino Rotary Encoder Tutorial

\*

\* by Dejan Nedelkovski, www.HowToMechatronics.com

\*

\*/

#define outputA 6

#define outputB 7

int counter = 0;

int aState;

int aLastState;

void setup() {

pinMode (outputA,INPUT);

pinMode (outputB,INPUT);

Serial.begin (9600);

// Reads the initial state of the outputA

aLastState = digitalRead(outputA);

}

void loop() {

aState = digitalRead(outputA); // Reads the "current" state of the outputA

// If the previous and the current state of the outputA are different, that means a Pulse has occured

if (aState != aLastState){

// If the outputB state is different to the outputA state, that means the encoder is rotating clockwise

if (digitalRead(outputB) != aState) {

counter ++;

} else {

counter --;

}

Serial.print("Position: ");

Serial.println(counter);

}

aLastState = aState; // Updates the previous state of the outputA with the current state

}

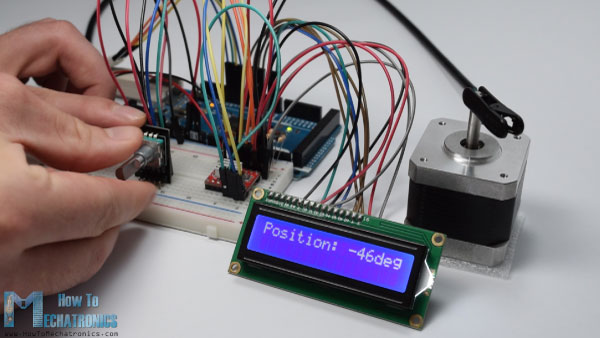
**Description of the code:** So first we need to define the pins to which our encoder is connected and define some variables needed for the program. In the setup section we need to define the two pins as inputs, start the serial communication for printing the results on the serial monitor, as well as read the initial value of the output A and put the value into the variable aLastState.

Then in the loop section we read the output A again but now we put the value into the aState variable. So if we rotate the encoder and a pulse is generated, these two values will differ and the first “if” statement will become true. Right after that using the second “if” statement we determine the rotation direction. If the output B state differ from the output A state the counter will be increased by one, else it will be decreased. At the end, after printing the results on the serial monitor, we need to update the aLastState variable with aState variable.

That’s all we need for this example. If upload the code, start the Serial Monitor and start rotating the encoder we will start getting the values in the serial monitor. The particular module that I have makes 30 counts each full cycle.

**Example 2 – Controlling a Stepper Motor Using a Rotary Encoder**

In addition to this basic example, I made one more example of controlling a [stepper motor](https://howtomechatronics.com/tutorials/arduino/how-to-control-stepper-motor-with-a4988-driver-and-arduino/) position using the rotary encoder.



Here’s the source code of this example:

/\* Stepper Motor using a Rotary Encoder

\*

\* by Dejan Nedelkovski, www.HowToMechatronics.com

\*

\*/

#include <LiquidCrystal.h> // includes the LiquidCrystal Library

LiquidCrystal lcd(1, 2, 4, 5, 6, 7); // Creates an LC object. Parameters: (rs, enable, d4, d5, d6, d7)

// defines pins numbers

#define stepPin 8

#define dirPin 9

#define outputA 10

#define outputB 11

int counter = 0;

int angle = 0;

int aState;

int aLastState;

void setup() {

// Sets the two pins as Outputs

pinMode(stepPin,OUTPUT);

pinMode(dirPin,OUTPUT);

pinMode (outputA,INPUT);

pinMode (outputB,INPUT);

aLastState = digitalRead(outputA);

lcd.begin(16,2); // Initializes the interface to the LCD screen, and specifies the dimensions (width and height) of the display }

}

void loop() {

aState = digitalRead(outputA);

if (aState != aLastState){

if (digitalRead(outputB) != aState) {

counter ++;

angle ++;

rotateCW();

}

else {

counter--;

angle --;

rotateCCW();

}

if (counter >=30 ) {

counter =0;

}

lcd.clear();

lcd.print("Position: ");

lcd.print(int(angle\*(-1.8)));

lcd.print("deg");

lcd.setCursor(0,0);

}

aLastState = aState;

}

void rotateCW() {

digitalWrite(dirPin,LOW);

digitalWrite(stepPin,HIGH);

delayMicroseconds(2000);

digitalWrite(stepPin,LOW);

delayMicroseconds(2000);

}

void rotateCCW() {

digitalWrite(dirPin,HIGH);

digitalWrite(stepPin,HIGH);

delayMicroseconds(2000);

digitalWrite(stepPin,LOW);

delayMicroseconds(2000);

}